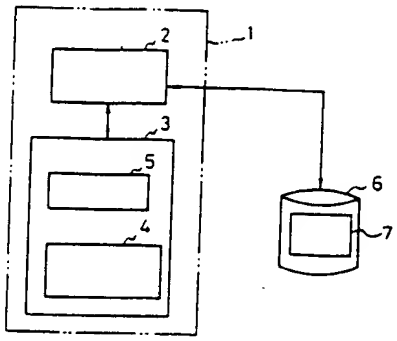
(54) **RESIDENT MODULE CORRECTING SYSTEM**
(11) **2-161523 (A)** (43) 21.6.1[redacted] (19) JP
(21) Appl. No. 63-317518 (22) 14.12[redacted]
(71) HITACHI LTD(1) (72) MASAAKI HAMA(1)
(51) **Int. Cl⁵. G06F9/06**

PURPOSE: To always discriminate whether each processing program module is being operated or not from an operation state managing table based upon each identification (ID) name by allocating each inherent ID name to each processing program module functionally closed to constitute an operating program residently stored in a main memory device.

CONSTITUTION: A program is stored in the main memory device 3 in a computer system 1 and data is processed by a processor 2 based upon the stored program. Each inherent ID name is allocated to each processing program module functionally closed to constitute an operating program stored in a program resident area 4 in the device 3 by an operating state control table 5 in the device 3 and whether each processing program module is being operated or not is always decided by the ID name. Whether the processing program module to be corrected is being operated or not is confirmed by the table and the module is corrected by the device 2.

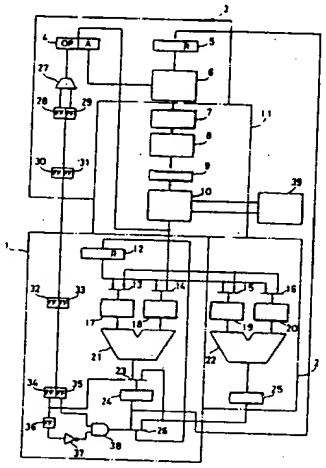


7: program library

(54) **ARITHMETIC PROCESSOR**
(11) **2-161525 (A)** (43) 21.6.1990 (19) JP
(21) Appl. No. 63-315566 (22) 14.12.1988
(71) NEC CORP (72) HIDESHI ISHII
(51) **Int. Cl⁵. G06F9/38**

PURPOSE: To eliminate the increment of hardwares by specifying whether the operated result of an instruction is used for the address modification of a succeeding instruction or not, and when the operated result is not used for the address modification, inhibiting the writing of the instruction in a register to be used for the address modification.

CONSTITUTION: Specified operation is executed by 1st and 2nd arithmetic units 1, 2 and an instruction is prefetched by an instruction prefetching unit 3. Operation data is stored in a general register 12 included in the unit 1 and a general register 5 in the unit 3 is used for the address modification of an instruction. An instruction word read out from a cache memory 10 is set up in an instruction register 4, an instruction code field in the register 4 is decoded by a decoder 27 and whether the operated result of the instruction stored in the unit 1 is to be used for the address modification of the succeeding instruction or not is specified by the contents of the instruction word.



6: address calculating circuit, 7: virtual address, 8: address converting circuit, 9: real address, 17 to 20: operand, 24,25: result, 39: main memory

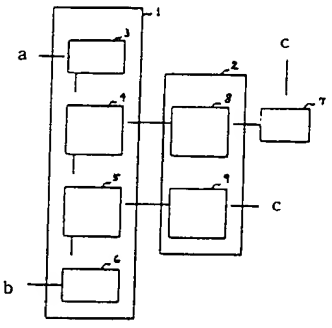
(54) **INFERENCE PROCESSOR**
(11) **2-161526 (A)** (43) 21.6.1990 (19) JP
(21) Appl. No. 63-315829 (22) 14.12.1988
(71) FUJITSU LTD (72) HIROTAKA HARA(1)
(51) **Int. Cl⁵. G06F9/44**

PURPOSE: To increase a rang to drive an inference engine by leading out a new opinion corresponding to an inputted opinion from the input opinion, and after adding the new opinion, applying an opinion/hypothesis type rule.

CONSTITUTION: An inference processor is constituted of the inference engine 1, a knowledge base 2, an opinion input part 3, an inter-opinion relation applying part 4, an opinion/hypothesis rule applying part 5, a hypothesis output part 6, a knowledge replacing part 7, an inter-opinion relation storing part 8, and an opinion/hypothesis storing part 9. In such a constitution, an opinion previously supported by the base 2 and an inter-opinion relation formula II (provided that NF_i: NOT of F_i) obtained by converting an opinion type rule formula I (provided that F_i is an opinion) are stored in the storage part 8. In the engine 1, inter-opinion relation generating contradictions in (n) NF_is among (n+1) NF_is constituting the opinion relation is extracted and the opinion regarding one residual NF_i as a real one is added by the application part 4. Then, the opinion/hypothesis rule is applied based upon the added result.

$$F_1 \wedge F_2 \wedge \cdots \wedge F_n \rightarrow F_0 \quad \text{I}$$

$$NF_1 \vee NF_2 \vee \cdots \vee NF_n \vee F_0 \quad \text{II}$$



a: observed opinion, b: defective hypothesis, c: opinion/hypothesis rule

⑲日本国特許庁(JP)　⑪特許出願公開

⑫公開特許公報(A)　平2−161523

⑤Int.Cl.⁵	識別記号	庁内整理番号	㊸公開　平成2年(1990)6月21日
G 06 F 9/06	440 F	7361−5B	

審査請求　未請求　請求項の数 2　(全6頁)

㊹発明の名称　常駐化モジュール修正方式

㉑特　願　昭63−317518
㉒出　願　昭63(1988)12月14日

㉗発明者　浜　正章　神奈川県横浜市中区尾上町6丁目81番地　日立ソフトウエアエンジニアリング株式会社内
㉗発明者　木村　伊九夫　神奈川県横浜市戸塚区戸塚町5030番地　株式会社日立製作所ソフトウエア工場内
㉑出願人　株式会社日立製作所　東京都千代田区神田駿河台4丁目6番地
㉑出願人　日立ソフトウエアエンジニアリング株式会社　神奈川県横浜市中区尾上町6丁目81番地
㉔代理人　弁理士　秋田　収喜

明細書

1．発明の名称

常駐化モジュール修正方式

2．特許請求の範囲

1．プログラムを記憶する主記憶装置と、記憶されたプログラムによりデータ処理を行う処理装置とを有する計算機システムにおいて、主記憶装置に常駐化され動作中のプログラムを構成する機能的に閉じた各処理プログラムモジュール毎に、それぞれ固有の識別名を付けて、常時、該識別名により各処理プログラムモジュールが動作中か否かを判断できる動作状態管理テーブルと、該動作状態管理テーブルにより、修正対象の処理プログラムモジュールが不動作中であることを確認し、処理プログラムモジュールを主記憶装置上で修正する手段とを有することを特徴とする常駐化モジュール修正方式。

2．動作状態管理テーブルは、処理識別名フィールド，動作カウンタフィールド，および修正表示フラグフィールドを有し、動作カウンタフィールドにより当該処理プログラムモジュールが処理動作中であることを指示し、修正表示フラグフィールドにより当該処理プログラムモジュールに対する修正動作中であることを指示することを特徴とする請求項1に記載の常駐化モジュール修正方式。

3．発明の詳細な説明

〔産業上の利用分野〕

本発明は、常駐化モジュール修正方式に関し、特に、記憶装置に常駐化されたプログラムによりデータ処理を行う処理装置を有する計算機システムにおいて、常駐化プログラムの処理プログラムモジュールをシステム稼動中にも修正できる常駐化モジュール修正方式に関するものである。

〔従来の技術〕

計算機システムにおいて、データ処理を行うプログラムはますます複雑化し、大型化している。このため、プログラムに不良が存在する確率が増大し、システム運用後におけるプログラム修正の機会が多くなっている。プログラム修正を行う方

法は、ソースプログラムに対して行う方法と、オブジェクトプログラムに対して行う方法とに大別できる。前者のソースプログラムに対して修正を行う方法は、修正後のソースプログラムを改めてコンパイルする必要があるため、短い時間で修正する必要がある場合は、後者のオブジェクトプログラムに対して修正する方法による方が便利である。

このようなオブジェクトプログラムに対する修正方法として、例えば、特公昭６３－１６２１号公報に記載されたようなオブジェクトプログラム修正方式がある。ここでのプログラム修正方式においては、オブジェクトプログラムの修正を次にようにして行う。まず、オブジェクトプログラムの修正を行うか否かの選択を行い、修正する場合に、修正対象のオブジェクトプログラムのローディングに先立って、プログラム修正情報を入力装置より主記憶装置に予め読み込む。その後、オブジェクトプログラムを主記憶装置にローディングする際に、そのオブジェクトプログラムをプログラム修正情報にしたがって主記憶装置上で修正する。

このようなオブジェクトプログラム修正方式によれば、システム立上げ時点から主記憶装置に常駐しないプログラムに対しては、該プログラムを主記憶装置に格納する時に、該プログラムを実行するためのジョブ制御文の指定にもとづき、該プログラムの命令語群の一部を主記憶装置上で修正することができる。しかし、システム立上げ時点から常駐化されるプログラムに対しては、システム立上げ以降のシステム稼動中には、主記憶装置上で該プログラムの命令語を直接修正できる方法ではない。

〔発明が解決しようとする課題〕

ところで、最近においては、計算機システムは長時間連続運転の必要性が高まり、計算機システムの２４時間連続（ノンストップ）運転形態がますます増える傾向にある。このため、システムを停止する機会が少なくなり、一旦、システムを停止して再立ち上げしなければ有効とならない常駐化システムプログラムに対するパッチ修正を、容易に行うことができないという問題点があった。

また、システム立上げ時点に主記憶装置に格納され、それ以降常駐化されるプログラムに対しても、システムを停止することなく、システム稼動中に主記憶装置上で該プログラムを直接修正することが要望される。

しかし、主記憶装置に常駐化しているプログラムの一部を修正するとき、プログラムの動作状態がチェックできず、修正対象の処理プログラムモジュールの動作中に、該処理プログラムモジュールを修正してしまいシステムに誤動作を生じることがあるので、常駐化しているプログラムを、主記憶装置上で直接修正することはできないという問題点があった。

本発明は、上記問題点を解決するためになされたものである。

本発明の目的は、主記憶装置に常駐化され動作中のプログラムに対しても、システム稼動中にシステムに誤動作を与えることなく、該プログラムの一部を主記憶装置上で直接修正することができる常駐化モジュールの修正方式を提供することにある。

本発明の前記ならびにその他の目的と新規な特徴は、本明細書の記述及び添付図面によって明らかになるであろう。

〔課題を解決するための手段〕

上記目的を達成するため、本発明においては、プログラムを記憶する主記憶装置と、記憶されたプログラムによりデータ処理を行う処理装置とを有する計算機システムにおいて、主記憶装置に常駐化され動作中のプログラムを構成する機能的に閉じた各処理プログラムモジュール毎に、それぞれ固有の識別名を付けて、常時、該識別名により各処理プログラムモジュールが動作中か否かを判断できる動作状態管理テーブルと、該動作状態管理テーブルにより、修正対象の処理プログラムモジュールが不動作中であることを確認し、処理プログラムモジュールを主記憶装置上で修正する手段とを有することを特徴とする。

また、動作状態管理テーブルは、処理識別名フィールド，動作カウンタフィールド，および修正表示フラグフィールドを有し、動作カウンタフィールドにより当該処理プログラムモジュールが処理動作中であることを指示し、修正表示フラグフィールドにより当該処理プログラムモジュールに対する修正動作中であることを指示することを特徴とする。

〔作用〕

前記手段によれば、主記憶装置に常駐化され動作中のプログラムを構成する機能的に閉じた各処理プログラムモジュール毎に、それぞれ固有の識別名を付けて、常時、該識別名により各処理プログラムモジュールが動作中か否かを判断できる管理テーブルを設ける。常駐化して動作するプログラムを、主記憶装置上で修正するとき、前記動作状態管理テーブルにより、修正対象の処理プログラムモジュールが不動作中であることを確認し、処理プログラムモジュールを主記憶装置上で修正する。

このように、各々の処理プログラムモジュールに対して、修正時点で動作中であると誤動作する可能性のある処理プログラムモジュールを識別名で指定し、該識別名の処理プログラムモジュールが動作中か否か判断する。そして、修正を行う場合、該修正時点で修正対象の処理プログラムモジュールが不動作中であることを確認し、当該修正対象の処理プログラムモジュールを主記憶装置上で修正を行う。

これにより、動作中に修正すると誤動作する可能性のある処理プログラムモジュールに対して、動作中でないときを確認して修正を行うことができ、システムを停止することなく、システムプログラムのパッチ修正を行うことができる。このため、システム稼動中であっても、主記憶装置に常駐化され動作するプログラムに対して、主記憶装置上で直接修正することができ、ソフトウエア保守のためのシステム停止の機会を少なくすることが可能となる。また、長時間連続運転されるシステムに対しても容易にプログラムのメンテナンス作業を行うことができる。

〔実施例〕

以下、本発明の一実施例を図面を用いて具体的に説明する。

第１図は、本発明の一実施例にかかる計算機システムの要部構成を示すブロック図である。第１図において、１は計算機システム、２は処理装置、３は主記憶装置である。主記憶装置３には、プログラム常駐化領域４と、常駐化したプログラムの各々の処理プログラムモジュールの動作状態を示す動作状態管理テーブル５とが設けられる。外部記憶装置６は、常駐化される各々のプログラムが格納されているプログラムライブラリ７をファイルとして格納しており、システム立上げ時に、プログラムライブラリ７から、常駐化プログラムが主記憶装置３にロードされて記憶され、以降システムの稼動中は、常駐化プログラムがプログラム常駐化領域４に常駐する。

第２図は、処理プログラムモジュールに対する動作状態管理テーブルの例を示す図である。第２図に示すように、動作状態管理テーブル５は、プログラムを構成する各処理プログラムモジュールの数の複数個のエントリから構成されており、各エントリには、各処理プログラムモジュールを識別するための処理識別名フィールド５ａ，当該処理プログラムモジュールが動作中か否かを示す動作カウンタフィールド５ｂ，および修正実行中表示フラグフィールド５ｃから構成されている。修正実行中表示フラグフィールド５ｃで表示する実行中表示フラグは、当該処理プログラムモジュールが主記憶装置上で修正実行中のときはオンとするフラグである。また、動作カウンタフィールド５ｂの動作カウンタは、当該処理プログラムモジュールの動作状態を表示するものであり、値が０のとき、動作中でないことを示している。

第３図は、常駐化プログラムを構成する各処理プログラムモジュールの処理フローを示すフローチャートである。第３図を参照して、各処理プログラムモジュールの処理の手順を説明する。

プログラムを構成する修正対象となる各処理プ

ログラムモジュールは、当該処理プログラムモジュールが修正中であれば、その実行を待たせなければならないため（修正中に実行すると誤動作するため）、まず、ステップ３１において、処理プログラムモジュールに対して、対応する処理識別名の修正実行中フラグがオフであるか否かを判定する。修正実行中表示フラグがオフでなければ、当該処理プログラムモジュールは、修正中であるので、ステップ３２に進み、所定の時間待ち処理を行い、再びステップ３１の判定処理を繰り返し行う。一方、ステップ３１の処理で、修正実行中表示フラグがオフと判定され、当該処理プログラムモジュールが修正中でなければ、ステップ３３において対応する処理識別名のエントリの動作カウンタをプラス１する。次にステップ３４で、当該処理プログラムモジュールによる処理を実行し、次のステップ３５において、当該処理プログラムモジュールによる処理の処理終了時点で、対応する処理識別名のエントリの動作カウンタをマイナス１して、処理を終了する。各々の処理プログラムモジュールは、このように処理されるので、動作のカウンタが０であれば、該処理プログラムモジュールは動作中でないことを示している。

常駐化され、上述のような処理の実行が行われいる各々の処理プログラムモジュールに対して、プログラムの一部に修正を行うパッチ処理を説明する。常駐化プログラムの修正指示パラメータは、例えば、次に示すような一連のプログラムステートメントにより与えられる。

```
WHEN  ALLOCPR=NO;
  THEN;
  DO;
        修正データ
  END;
```

このプログラムステートメントの例は、処理識別名ＡＬＬＯＰＲが動作中でないとき、修正データで示す修正を行うプログラムを示している。

第４図は、常駐化プログラムを構成する各処理プログラムモジュールに対する修正処理の処理フローを示すフローチャートである。第４図を参照して、常駐化プログラムに対する修正処理の処理手順を説明する。まず、最初に、ステップ４１において、外部記憶装置上のプログラムライブラリに格納されている該当プログラムの修正処理を行う。次に、ステップ４２において、次に修正指示パラメータで指定された処理識別名を持つ処理プログラムモジュールが動作中か否かを動作カウンタの値で判定する。すなわち、修正対象の指定の処理識別名の動作カウンタは０か否かを判定し、動作カウンタの値が０でなければ、当該処理プログラムモジュールを用いている動作中プロセスが存在するため、ステップ４３に進み、所定の待ち時間処理を行い、再び、ステップ４２の判定処理を行う。ステップ４２の判定処理で、動作カウンタが０であることが判定されると、当該処理プログラムモジュールは動作中でないため、修正データにしたがって、主記憶装置上で当該プログラムの修正処理を行うことができる。この修正処理においては、修正処理の実行中に修正対象の処理プログラムモジュールが動作するのを防ぐため、ステップ４４において、当該処理プログラムモジュールの修正実行中表示フラグをオンとし、次のステップ４５において、修正データにしたがい、主記憶装置上で当該プログラムを修正する処理を行う。修正処理の終了後、次のステップ４６において、修正対象の処理プログラムモジュールに対する修正実行中表示フラグをオフとして、処理を終了する。

このようにして修正処理を行うことにより、修正対象の処理プログラムモジュールが動作中でないときを確認して、修正対象の処理プログラムモジュールを修正することができ、処理プログラムモジュールの修正処理と、修正対象の当該処理プログラムモジュールによる処理の動作が同時に実行されることはない。

以上、本発明を実施例にもとづき具体的に説明したが、本発明は、前記実施例に限定されるものではなく、その要旨を逸脱しない範囲において種々変更可能であることは言うまでもない。

〔発明の効果〕

以上、説明したように、本発明によれば、動作中に修正すると誤動作する可能性のある処理プログラムモジュールに対して、動作中でないときを確認して修正を行うことができ、システムを停止することなく、システムプログラムのパッチ修正を行うことができる。このため、システム稼動中であっても、主記憶装置に常駐化され動作するプログラムに対して、主記憶装置上で直接修正することができ、ソフトウエア保守のためのシステム停止の機会を少なくすることが可能となる。また、長時間連続運転されるシステムに対しても容易にプログラムのメンテナンス作業を行うことができる。

4．図面の簡単な説明

第1図は、本発明の一実施例にかかる計算機システムの要部構成を示すブロック図。

第2図は、処理プログラムモジュールに対する動作状態管理テーブルの例を示す図。

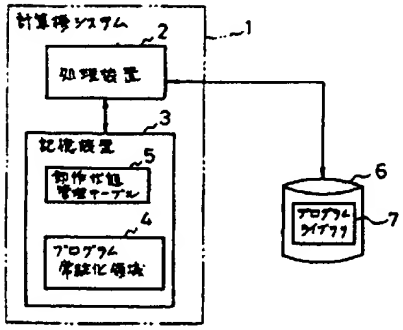
第3図は、常駐化プログラムを構成する各処理プログラムモジュールの処理フローを示すフローチャート。

第4図は、常駐化プログラムを構成する各処理プログラムモジュールに対する修正処理の処理フローを示すフローチャートである。

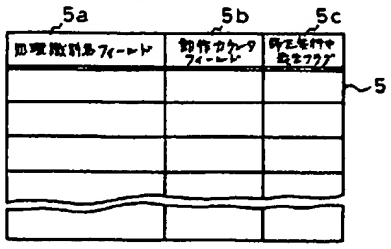
図中、1…計算機システム、2…処理装置、3…主記憶装置、4…プログラム常駐化領域、5…動作状態管理テーブル。

代理人　弁理士　秋田収喜

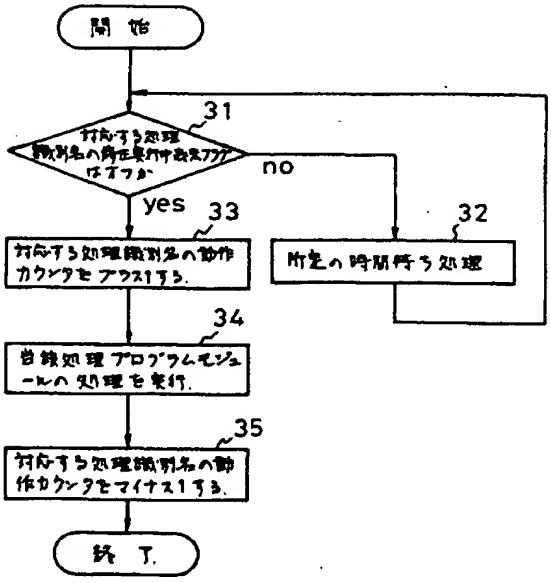
第1図



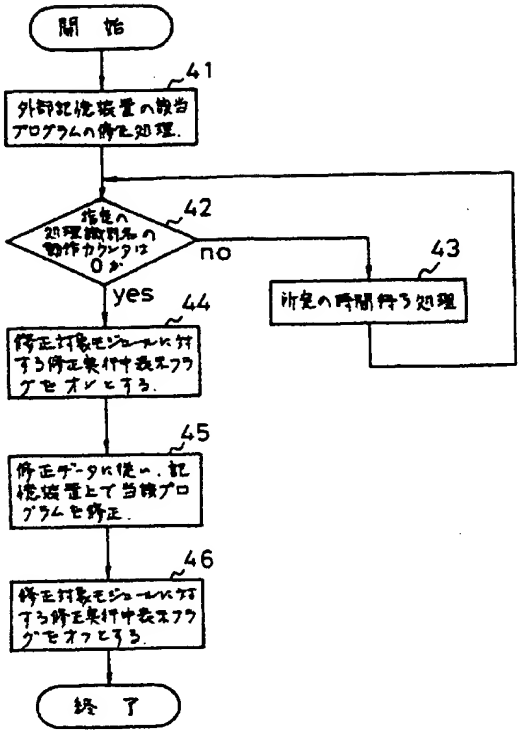
第2図



第3図



第4図


開始
41
外部記憶装置の割当プログラムの修正処理.
42
指定の処理装置名の動作カウンタは0か
no
43
所定の時間待つ処理
yes
44
修正対象モジュールに対する修正実行中表示フラグをオンとする.
45
修正データに従い、記憶装置上で当該プログラムを修正.
46
修正対象モジュールに対する修正実行中表示フラグをオフとする.
終了